Vietnam J Comput Sci (2014) 1:179–191
DOI 10.1007/s40595-014-0020-y

REGULAR PAPER

Adaptive reversible data hiding with pyramidal structure

Yuh-Yih Lu · Hsiang-Cheh Huang

Received: 30 November 2013 / Accepted: 26 March 2014 / Published online: 24 April 2014

Abstract In this paper, we propose an adaptive algorithm for reversible data hiding by employing the characteristics and pyramidal relationships of original images. The major goal of reversible data hiding is to keep the reversibility of algorithm. By use of the pyramidal structure to explore the inherent characteristics of original images, regions with different smoothness levels can be determined, and then data hiding can be performed adaptively with the pre-determined threshold for balancing the output image quality and embedding capacity. On the one hand, larger capacity can be hidden into smoother regions with limited degradation of output image quality. On the other hand, the size of location map, which serves as the side information for keeping reversibility, can be reduced for embedding into smoother or less smooth regions of original image. By carefully manipulating difference values between layers in pyramidal structure, secret information can effectively be embedded. With our method, we observe better performances over relating methods with enhanced image quality, the more embedding capacity, and comparable amount of side information for decoding. More importantly, the reversibility of our method is guaranteed, meaning that original image and secret information can both be perfectly recovered at the decoder. Simulation results demonstrate that proposed method in this paper outperforms those in conventional algorithms.

Y.-Y. Lu
Minghsin University of Science and Technology, No.1, Xinxing Rd., Xinfeng, Hsinchu 30401, Taiwan, R.O.C.
e-mail: yylu@must.edu.tw

H.-C. Huang (✉)
National University of Kaohsiung, No. 700 University Road, Kaohsiung 811, Taiwan, R.O.C.
e-mail: huang.hc@gmail.com
URL: http://sites.google.com/site/hch888dr/

Keywords Reversible data hiding · Pyramidal structure · Histogram · Quad · Image quality · Capacity

1 Introduction

Information security is one of the popular research topics, and it is also an important issue for practical application. Among relating methods in information security and corresponding digital rights management (DRM) systems [1,2], cryptography and watermarking are two important categories. We focus on reversible data hiding algorithm in this paper, which belongs to a branch in watermarking researches and applications.

Watermarking researches have emerged for around 15 years, and reversible data hiding is a recently developed branch in watermarking researches [3,4]. For conventional watermarking, at the encoder, the secret information should be embedded into the original multimedia contents, digital images in most cases, by the use of algorithms developed by researchers. Then, the watermarked media can be transmitted to the receiver. Data loss or intentional attacks may be experienced during transmission. After reception of the delivered watermarked media, only the secret information needs to be extracted [1]. In contrast, for reversible data hiding, data embedding is similar to its counterpart with conventional watermarking applications. Different from watermarking, for reversible data hiding, after the reception of marked media, both the original content and embedded secret information need to be recovered and extracted perfectly with a reasonable amount of side information [5,6]. And this is the origin of the term "reversible" comes from. Besides the development of algorithms, reversible data hiding can be applicable to the protection of medical images [7,8], or the integration with encryption techniques [9]. Due to this kind of character-

istics, during the transmission, the watermarked media need to be kept intact.

Suppose that there are lots of medical images in the database of some hospital. Due to the stressful environment, especially in ICU, doctors or nurses may unintentionally put Patient A's personal data and medical records into Patient B's images. With the aid of reversible data hiding, Patient A's medical records can be embedded into Patient A's images beforehand [7]. For the doctors and nurses, while retrieving patients' marked images, corresponding medical records can also be extracted to compare to the database. Also, original images can be perfectly recovered to meet the integrity. Should there be any mismatch, doctors or nurses are alarmed to prevent anything unexpected from happening. Thus, reversible data-hiding techniques can be applicable for practical use.

For evaluating performances of algorithms, and for making fair comparisons, parameters from different aspects should be considered. These parameters include the following.

- *Reversibility* it implies that marked image should be decomposed into original image and secret information perfectly at the decoder.
- *Output image quality, or imperceptibility* it denotes the resemblance between the original and output images, meaning that the error induced from data embedding should be as small as possible.
- *Capacity* it means the number of bits that can be embedded in the original image, which is expected to be larger than some reasonable amount. Larger capacity provides the flexibility for the selection of secret information, however, larger degradation may be expected correspondingly.
- *Side information, or the overhead for decoding* it should be as little as possible to make the proposed algorithm suitable for practical applications.

As far as we know, considering practical implementations, some tradeoffs among the parameters should be watched for the design of algorithm. For instance, embedding more capacity into original image introduces larger error, hence the degradation of quality of marked image. We suggest choosing the two criteria of obtaining at least 1.0 bit/pixel (bpp) of maximal embedding capacity, and reaching at least 30 dB in peak signal-to-noise ratio (PSNR) of output image quality. With our algorithm, reversible data hiding can be reached with adaptive embedding and pyramidal structure based on parameters listed above. Reversible data-hiding methods, which will be described in Sect. 2, have their inherent limitations and drawbacks even though lots of advantages can be observed. More importantly, few methods take the characteristics of original images into account in this field. Here, we make use of pyramidal structure of original image for obtaining the larger number of secret bits for embedding, with similar quality of the output images. Simulation results reveal that the algorithm proposed in this paper outperforms conventional ones by use of eight test images.

This paper is organized as follows. In Sect. 2, we describe fundamental concepts of reversible data hiding algorithms, including the histogram-based and difference expansion (DE)-based schemes. The reason why reversibility can be guaranteed is also addressed. Then, in Sect. 3, by considering inherent characteristics of images, we can utilize the difference values, and present the better way to make use of the pyramidal structure for reversible data hiding. Simulation results are demonstrated in Sect. 4, which point out the guaranteed image quality, the more embedding capacity, and the less side information needed for the proposed algorithm. Finally, we conclude this paper in Sect. 5.

2 Implementations for reversible data hiding

The framework of reversible data hiding can be demonstrated in Fig. 1. On the one hand, in Fig. 1a, it depicts the encoder framework. Original image and secret information are integrated altogether with the devised algorithm to form the marked image. For keeping reversibility, the necessary amount of side information should also be provided to the

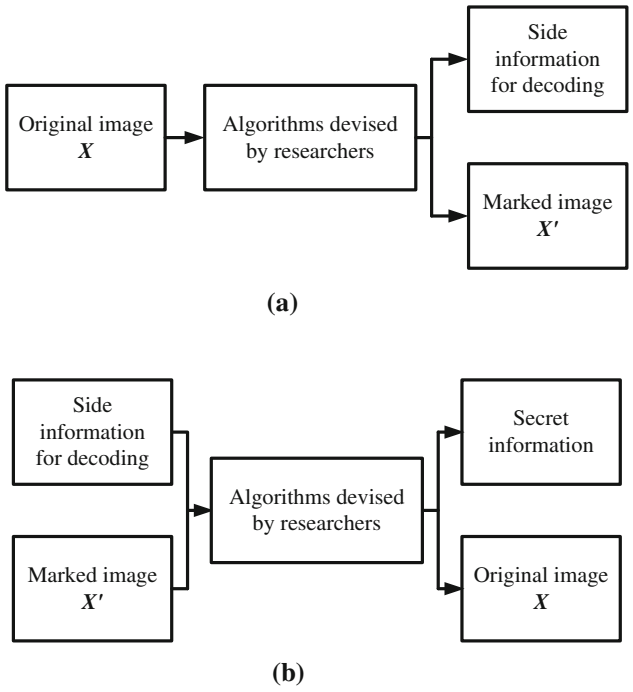


Fig. 1 Framework of reversible data hiding. **a** Encoder framework. **b** Decoder framework

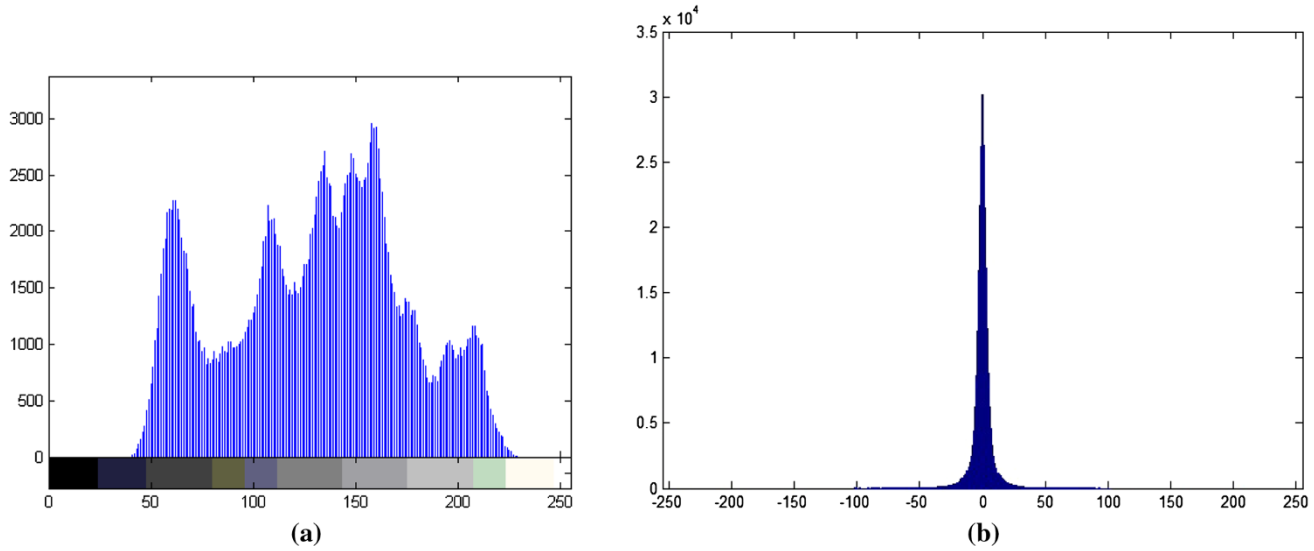


Fig. 2 Comparisons of histogram and difference histogram with Lena. **a** Histogram of Lena, H, with the peak occurrence of 2,966. **b** Difference histogram of Lena, D, with the peak occurrence of 30,150

decoder. On the other hand, in Fig. 1b, it displays the decoder framework. It is easily observed that blocks in Fig. 1b are placed in reverse order comparing to its counterpart in Fig. 1a. By doing so, both the original image and secret information can be perfectly separated from the marked image with the devised algorithm. And this is the major reason about the name of "reversible data hiding".

Practical implementations for making reversible data-hiding possible can roughly be categorized into two major branches. From global point of view, by carefully modifying the histogram, we can reversibly embed the secret information into original image with schemes in [10–12]. Schemes in this branch are referred to as the histogram-based schemes. On the other hand, considering local characteristics of original image, we can embed secret information by intentionally doubling the difference value between neighboring pixel pairs with schemes in [13–15]. Schemes in this branch are referred to as the DE-based schemes.

Here, we briefly address the advantages and drawbacks of both schemes. First, for the histogram-based schemes, it has the advantage of guaranteed output image quality because the mean square error (MSE) between the marked and original image is limited to be below 1, leading to the result of at least 48.13 dB in PSNR value [3]. The major drawback of histogram-based schemes is the limited number of capacity, which is constrained by the peak of the histogram.

Next, for the DE-based schemes, it utilizes the difference value between two neighboring pixels for embedding one secret bit, leading to the capacity of 0.5 bit/pixel (bpp). However, after modifying the difference values, it may cause the overflow for producing the marked image. By following [5] and [6], the side information is named 'location map' (LM), which should be recorded in advance to keep the reversibility. There are some effective means for reducing the size of LM in [13] and [14]. Besides, unlike the histogram-based schemes, output image quality cannot be guaranteed.

It may be constructive to integrate the two schemes altogether and to acquire the advantages from both schemes. We take the histogram H in Fig. 2a, and difference histogram D in Fig. 2b of test image Lena with size of 512×512. With the 8-bit grey-level representation, the pixel values are integers between 0 and 255. Consequently, the range of difference values lies between -255 and 255. We observe that the peak values of Fig. 2a and b are 2,966 and 30,150, respectively, which results in 10.17 times difference. If we can borrow the concept in histogram-based schemes in Fig. 2a and integrate into DE-based scheme in Fig. 2b, larger capacity may be expected by utilizing the difference histogram. Output image quality can also be controlled when the limited amount of capacity is embedded.

Here is a simple illustration for reversible data hiding with difference histogram. The difference histogram can be produced from the difference between neighboring pixels. In Fig. 2b, we observe the difference histogram D is concentrated around 0. Here, D is an array, and we can denote the array by $D = [d[-255], d[-254], \ldots, d[-1], d[0], d[1], \ldots, d[254], d[255]]$, because the difference values lie between -255 and 255. Next, the predetermined threshold value δ, which is a positive integer, is selected for data embedding, and it also serves as the side information at the decoder. For embedding the secret information, the altered difference histogram D' should be formed first. By following the same manner, D' can be represented with the notation of $D' = [d'[-255], d'[-254], \ldots, d'[-1], d'[0], d'[1], \ldots,$

$d'[254], d'[255]]$. Next, data embedding should meet one of the following cases.

Case 1. For $d[i], \quad i \geq \delta + 1$,

$$d'[i+1] = d[i]. \tag{1}$$

Case 2. For $d[i], \quad i \leq -\delta$,

$$d'[i-1] = d[i]. \tag{2}$$

Case 3. For $d[i], \quad -\delta + 2 \leq i \leq \delta - 1$, the values are kept the same. That is,

$$d'[i] = d[i]. \tag{3}$$

Case 4. For $i = -\delta + 1$ and $i = \delta$, the values are intentionally set to 0. That is,

$$d'[-\delta + 1] = d'[\delta] = 0. \tag{4}$$

We observe that the value of δ plays the role of the secret key in reversible data hiding with only a few bits of overhead. It has another advantage of ease of implementation because only the moving of some portion of difference histogram is needed, and there is no need for calculation. Besides the advantages indicated above, there is one drawback for the proposed algorithm. Under the extreme cases when the index i reaches -255 or 255, the overflow problem would occur, which can be easily observed from Eqs. (1) and (2). Such locations, or LM, should be recorded and served as the side information for decoding.

From Case 1 to Case 4, histogram occurrences at two difference values of δ and $(-\delta + 1)$, or the two bins as described in Case 4, are intentionally set to zero for hiding bit '0' and bit '1'. For embedding secret bits, the difference histogram containing the secret, D'', should be produced correspondingly. Again, by following the same manner, D'' can be represented with the notation of $D'' = [d''[-255], d''[-254], \ldots, d''[-1], d''[0], d''[1], \ldots, d''[254], d''[255]]$. For clarity, four the difference values (or the index i) at $-\delta$, $(-\delta + 1)$, δ, and $(\delta + 1)$ are employed for data embedding. Other elements in D'' are identical to their corresponding counterparts in D'. Embedding meets one of following conditions at the encoder.

- For embedding bit '1': for positive difference,

$$d''[\delta] = d'[\delta + 1]. \tag{5a}$$

For negative difference,

$$d''[-\delta + 1] = d'[-\delta]. \tag{5b}$$

- For embedding bit '0', keep the difference values the same. That is,

$$d''[\delta + 1] = d'[\delta + 1]. \tag{6a}$$

$$d''[-\delta] = d'[-\delta]. \tag{6b}$$

The difference values for remaining elements in D'' are identical to their corresponding counterparts in D'.

If we look into more detail in Eqs. (5a) and (5b), addition or subtraction by 1 implies the embedding of one bit. It has the potential to add or subtract the value of $2^n - 1$, with n being the number of secret bits, for data embedding. For instance, if $n = 2$, addition or subtraction the difference values by 0–3 is able to hide two bits at the same time. Meanwhile, for the locations of difference values larger than 252 or smaller than -252, they should be recorded as LM. It corresponds to the observation that for smoother regions, they have the potential to hide more bits simultaneously. Larger value of n, or embedding more bits at the same time, might be impractical because the added or subtracted value grows exponentially, which implies the increased amount of LM. By doing so, adaptive embedding can be achieved by incorporating with secret size and smoothness of original image.

For the extraction of secret bits and the recovery of original image, they correspond to the reverse procedures to data embedding, as depicted in the framework in Fig. 1. They can be described with the following procedures.

1. The side information, which includes LM and δ value, along with the marked image, should be obtained at the decoder.
2. Then, difference histogram containing secret bits D'' can be produced from marked image.
3. With the δ value, secret bits of 0 and 1 can be extracted from D'' with Eqs. (5a) and (6a). Next, D' can be recovered after the extraction of secret bits.
4. In D', remove the empty bins at $d'[-\delta + 1]$ and $d'[\delta]$. By adding back the extremes of $d[-255]$ and $d[255]$ from LM, original difference histogram D can be formed.
5. Recover the original image by adding the difference value back to the seed pixel.

With the descriptions above, we can find that reversibility can be guaranteed by manipulating difference histogram for reversible data hiding.

3 Proposed algorithm

We propose our algorithm by considering the three-tier procedures with the concepts described in Sect. 2. The difference

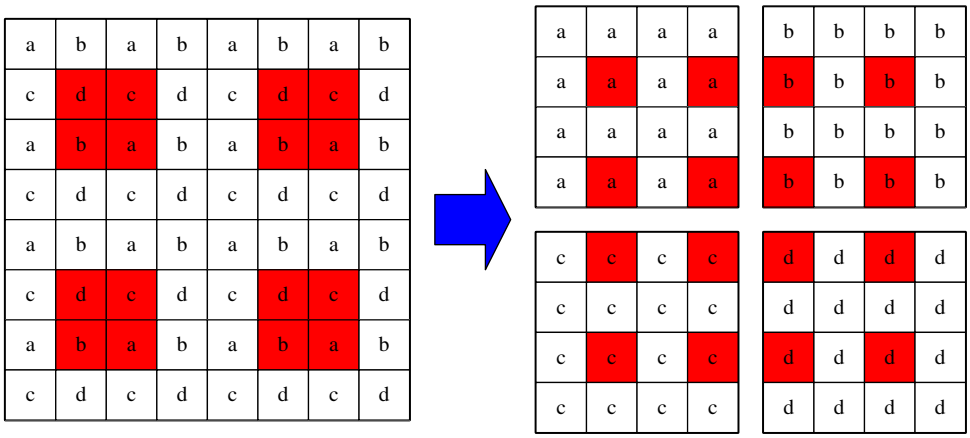


Fig. 3 The splitting of original image. Pixels in *red* in the *left image* are prepared for pyramidal structure. The image at the *right* side corresponds to the result after splitting

values between neighboring pixels, as well as the pyramidal structure, are utilized to look for better performances. As we mentioned in Sect. 1, for our algorithm, we suggest reaching the performances of at least 1.0 bpp of capacity, and at least 30 dB in PSNR of output image quality. After looking for major research databases, two relating papers [16,17] met the two criteria, and they are employed to make comparisons with proposed algorithm.

3.1 Tier #1: splitting of original image

By making good use of the characteristics of original image, we first divide the original image $\mathbf{X}$ into non-overlapping 2×2 blocks, and each block corresponds to one quad. In order to look for the reduction of side information to be provided to the receiver, and to make good use of difference values calculated from each quad, we choose regular pattern to serve as reference pixels for reversible data hiding.

For the ease of demonstration, we split the original image $\mathbf{X}$ into non-overlapping groups, and each group is composed of pixels from positions in 'a', 'b', 'c', and 'd', shown in the left part of Fig. 3. Next, we gather pixels in 'a', 'b', 'c', and 'd' together to form the sub-images of $\mathbf{X}_a$, $\mathbf{X}_b$, $\mathbf{X}_c$, and $\mathbf{X}_d$, respectively, depicted in the right part of Fig. 3. Each square block represents one pixel in the image.

Let the pixels in red serve as the reference points for data hiding. Because they are placed on regular positions, the side information for decoding may be reduced. We can use two bits to present the four types of positions of 'a', 'b', 'c', and 'd'. In addition, the arrangements of red pixel positions may associate with hierarchical coding, or layered coding, where the original image and the pixels in red may serve as the base layer and enhancement layer, respectively. For instance, we can gather pixels in red in the left part of Fig. 3 altogether to form a smaller image corresponding to the original image. We can carefully utilize the relationships between base and enhancement layers to look for better performances in reversible data hiding.

3.2 Tier #2: multi-level embedding of secret information

With the four split sub-images and the reference points, data embedding can be performed accordingly by following the concepts described in Sect. 2.

In each of the sub-images, we first divide the image into non-overlapping quads. We can observe the arrangement of one reference point (or the pixel in red) in one quad. We take the first quad in $\mathbf{X}_a$ as an instance in Fig. 3 for the better comprehension of our method. For other sub-images, by replacing the place of reference points, same steps can be performed subsequently. Pixels in this quad locate at $X_a(1,\ 1)$, $X_a(1,\ 2)$, $X_a(2,\ 1)$, and $X_a(2,\ 2)$, and the reference point, shown in red, is $X_a(2,\ 2)$. The luminance of the reference is kept unchanged, and three difference values can be calculated with the following equations:

$$d_1 = \mathrm{lum}(X_a(1,\ 1)) - \mathrm{lum}(X_a(2,\ 2)); \tag{7a}$$

$$d_2 = \mathrm{lum}(X_a(1,\ 2)) - \mathrm{lum}(X_a(2,\ 2)); \tag{7b}$$

$$d_3 = \mathrm{lum}(X_a(2,\ 1)) - \mathrm{lum}(X_a(2,\ 2)); \tag{7c}$$

Next, by following the methods in [16] and [17] with some modifications, based on the concept depicted in Eqs. (5a)–(6a), a predetermined threshold T, which relates to the embedding strength, should be compared with the difference values. Because the maximum of difference may be close to the threshold value T, overflow may occur, which would lead to the difficulty to keep reversibility of algorithm. Steps for performing data hiding can be executed as follows:

Step 1. If $\max(|d_1|,\ |d_2|,\ |d_3|) < \frac{1}{8}T$, two bits can be embedded, which are represented by b_1b_2, with $b_1,\ b_2 \in \{0,\ 1\}$. The difference value is modified by

$$d_i' = 4 \cdot d_i + b_1b_2, \quad i = 1,\ 2,\ 3. \tag{8}$$

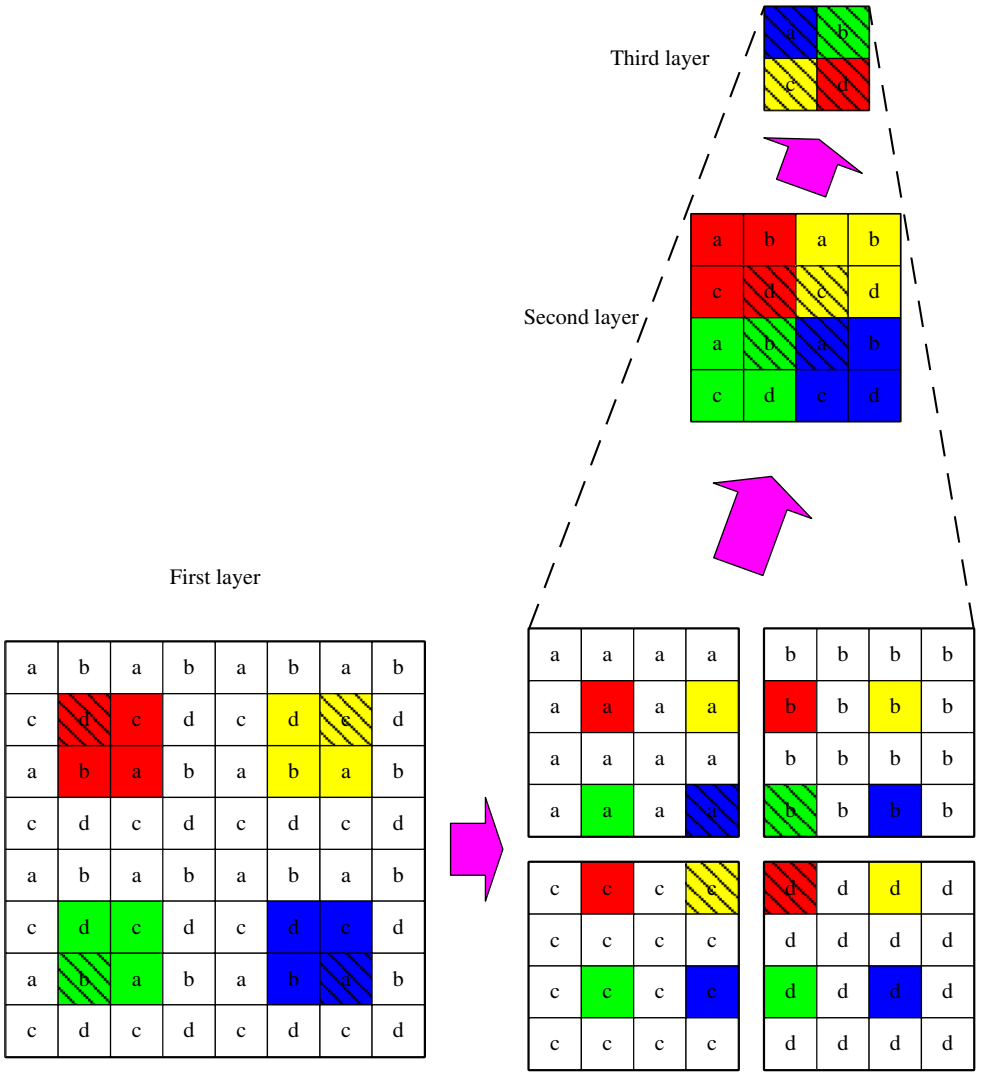


Fig. 4 The pyramid structure. The reference points in Fig. 3 are denoted in the *left*, and gather them to become a quarter-sized image. By following this manner, pyramidal structure can be formed

The two secret bits, $b_1b_2 \in \{00,\ 01,\ 10,\ 11\}$, are concatenated together for embedding at the same time. Decimal forms of b_1b_2 are expected for the modification of difference values as depicted in Eq. (8). Because the difference values are much smaller than T, a total of six bits can be embedded into a quad based on Eqs. (7a)– (7c), leading to the capacity of $\frac{6}{4} = 1.5$ bit/pixel (bpp).

Step 2. If $\frac{1}{8}T \le \max(|d_1|,\ |d_2|,\ |d_3|) < \frac{1}{2}T$, one bit can be embedded, which is represented by b, with $b \in \{0,\ 1\}$. The difference value is modified by

$$d_i' = 2 \cdot d_i + b, \quad i = 1,\ 2,\ 3. \tag{9}$$

In Eq. (9), b denotes the secret bit. By doing so, three bits can be embedded into a quad, leading to the capacity of $\frac{3}{4} = 0.75$ bpp.

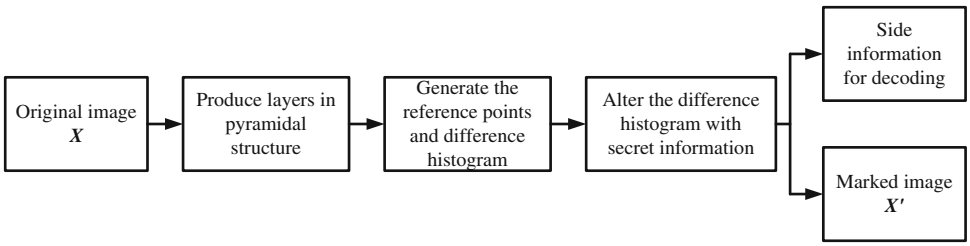
Step 3. If $\frac{1}{2}T \le \max(|d_1|,\ |d_2|,\ |d_3|) < T$, one bit can be embedded. The difference value is modified by

$$d_i' = 2 \cdot \left\lfloor \frac{d_i}{2} \right\rfloor + b, \quad i = 1,\ 2,\ 3. \tag{10}$$

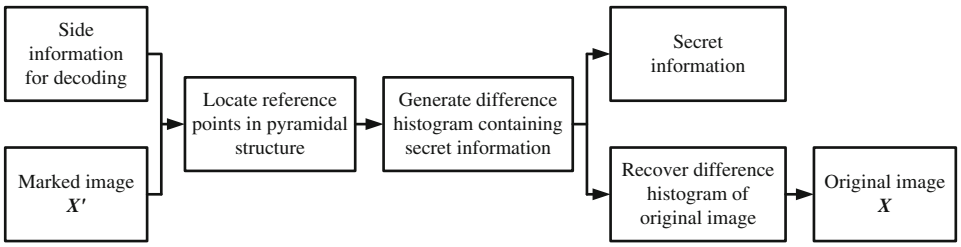
In Eq. (10), the new difference value d_i' is produced by changing the least significant bit of d_i, the symbol $\lfloor \bullet \rfloor$ means the floor function, b denotes the secret bit, and the capacity of 0.75 bpp can be reached.

Step 4. If $\max(|d_1|,\ |d_2|,\ |d_3|) \ge T$, no bit can be embedded because the difference becomes too large to become unsuitable for embedding. All the values in the quad are kept unchanged.

In order to avoid possible decoding errors, the four steps are recorded with the two-bit side information for correct decoding at the receiver. After performing one of the above four steps, the new difference value d_i' is added back with the luminance of the reference point, $X_a(2,\ 2)$. The new luminance values in the first quad of $\mathbf{X}_a$ can be calculated as

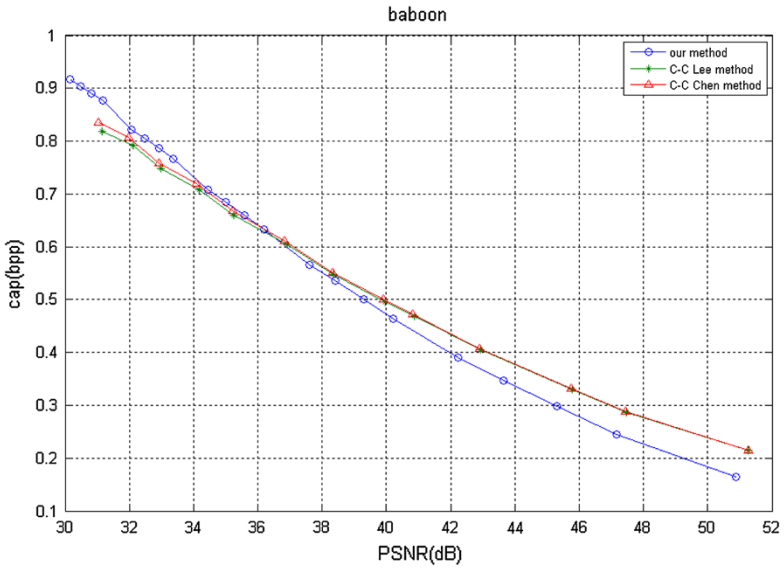


(a)

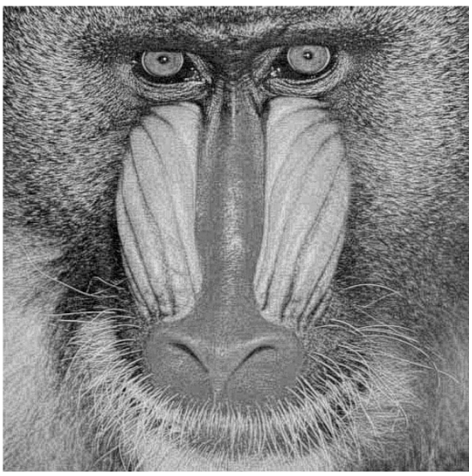


(b)

Fig. 5 Framework of proposed algorithm. **a** Encoder framework. **b** Decoder framework



(a)



(b)

Fig. 6 Results with `baboon`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 260,679 bits (0.9944 bpp). Embedding strength is 48, and PSNR value is 30.1571 dB

follows.

$$X'_a(1,\ 1) = d'_1 + X_a(2,\ 2); \tag{11a}$$

$$X'_a(1,\ 2) = d'_2 + X_a(2,\ 2); \tag{11b}$$

$$X'_a(2,\ 1) = d'_3 + X_a(2,\ 2); \tag{11c}$$

$$X'_a(2,\ 2) = X_a(2,\ 2); \tag{11d}$$

With the operation in Eqs. (11a)–(11d), four sub-images containing hidden information, or $\mathbf{X}'_a$, $\mathbf{X}'_b$, $\mathbf{X}'_c$, and $\mathbf{X}'_d$, can be formed. Finally, by following the reverse operation to Fig. 3, the output image $\mathbf{X}'$ can be produced.

For the decoder, reverse steps can be performed accordingly, with the following steps. Steps for performing data extraction and recovery of original can be executed as follows:

Step 1. The image containing hidden secret, $\mathbf{X}'$, should be split using the method in Fig. 3.

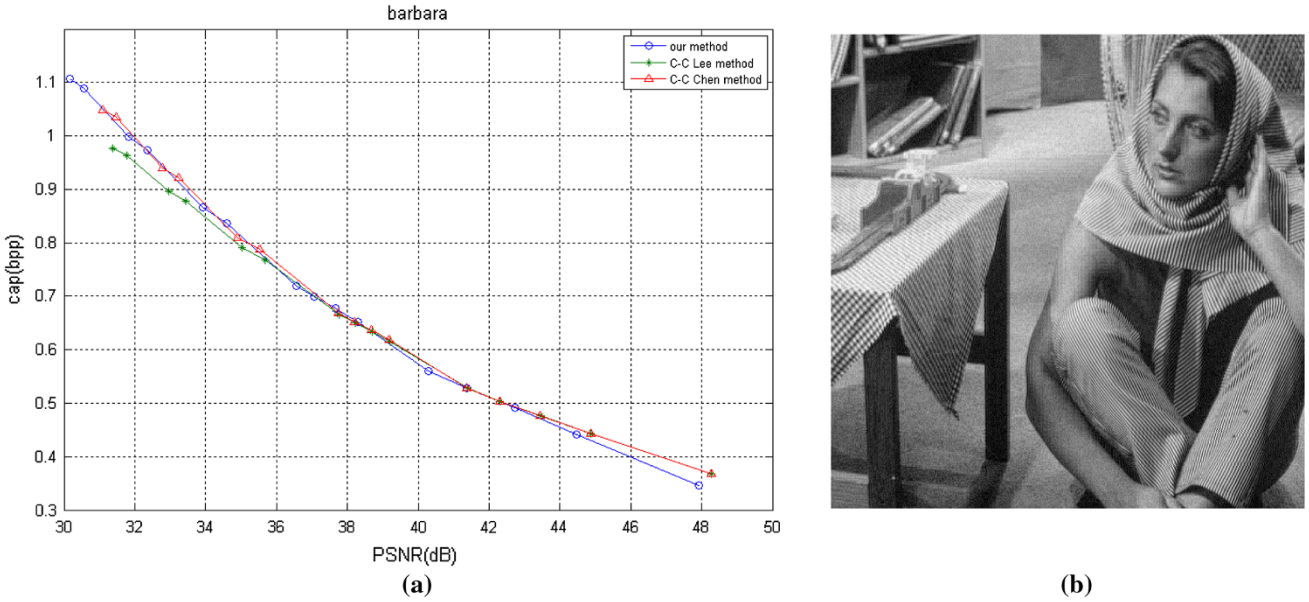


Fig. 7 Results with `Barbara`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 310,653 bits (1.1850 bpp). Embedding strength is 48, and PSNR value is 30.1648 dB

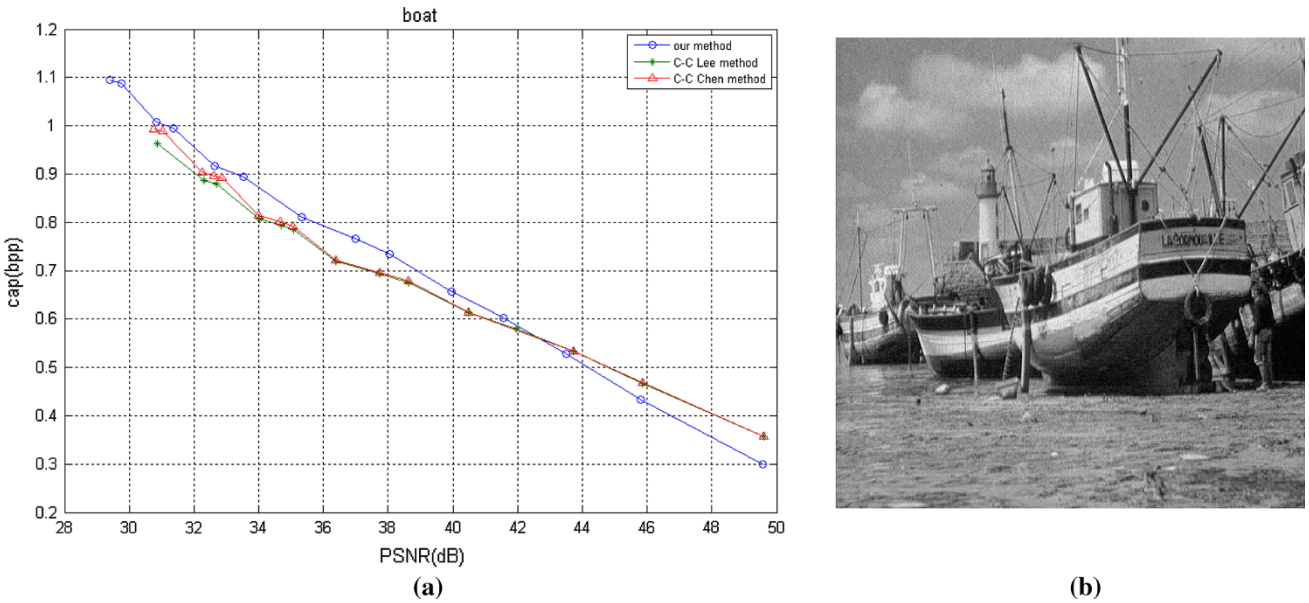


Fig. 8 Results with `boat`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 307,743 bits (1.1739 bpp). Embedding strength is 48, and PSNR value is 29.3969 dB

Step 2. For every quad, the difference values are calculated with Eq. (7) based on the reference points.

$$d'_1 = X'_a(1,\ 1) - X_a(2,\ 2); \tag{12a}$$

$$d'_2 = X'_a(1,\ 2) - X_a(2,\ 2); \tag{12b}$$

$$d'_3 = X'_a(2,\ 1) - X_a(2,\ 2); \tag{12c}$$

Step 3. With the prespecified threshold value T, hidden secret can be extracted based on Eqs. (8), (9), or (10). Original difference values can also be acquired simultaneously.

Step 4. Recover the original image $\mathbf{X}$ with the original difference values and the luminance of reference points.

3.3 Tier #3: employing the pyramidal structure

With the methods in Sect. 3.2, we can further perform data hiding with the pyramidal structure based on the reference points. From the depiction in Fig. 4, we take the three-layer

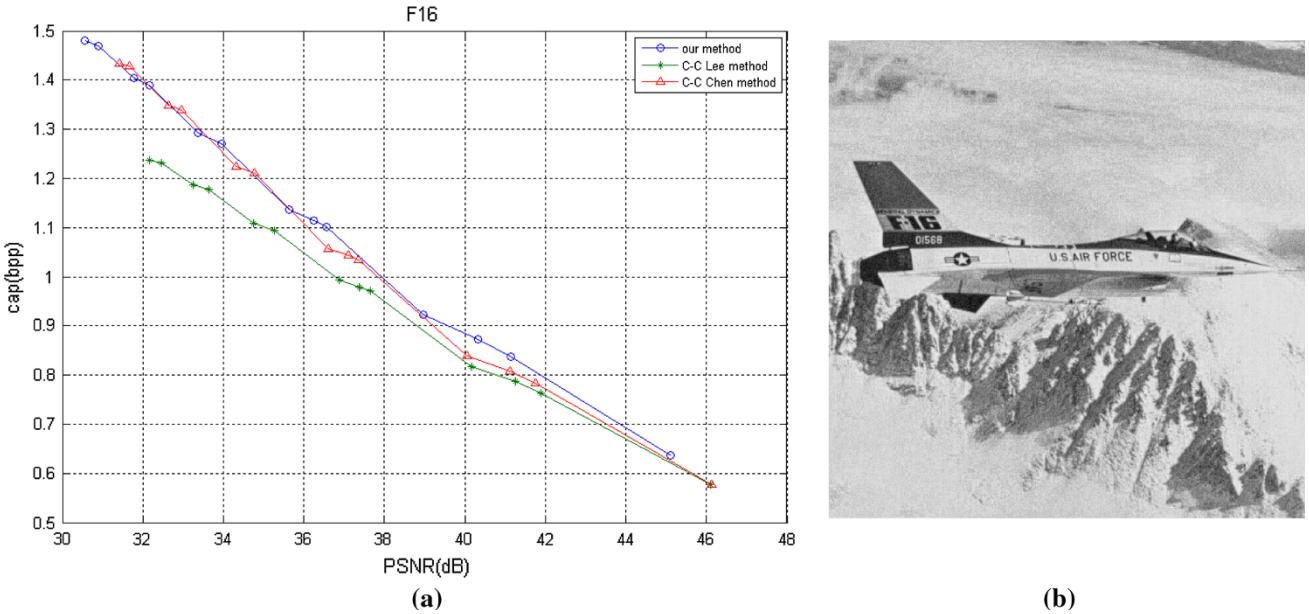


Fig. 9 Results with `F16`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 408,414 bits (1.5580 bpp). Embedding strength is 48, and PSNR value is 30.5545 dB

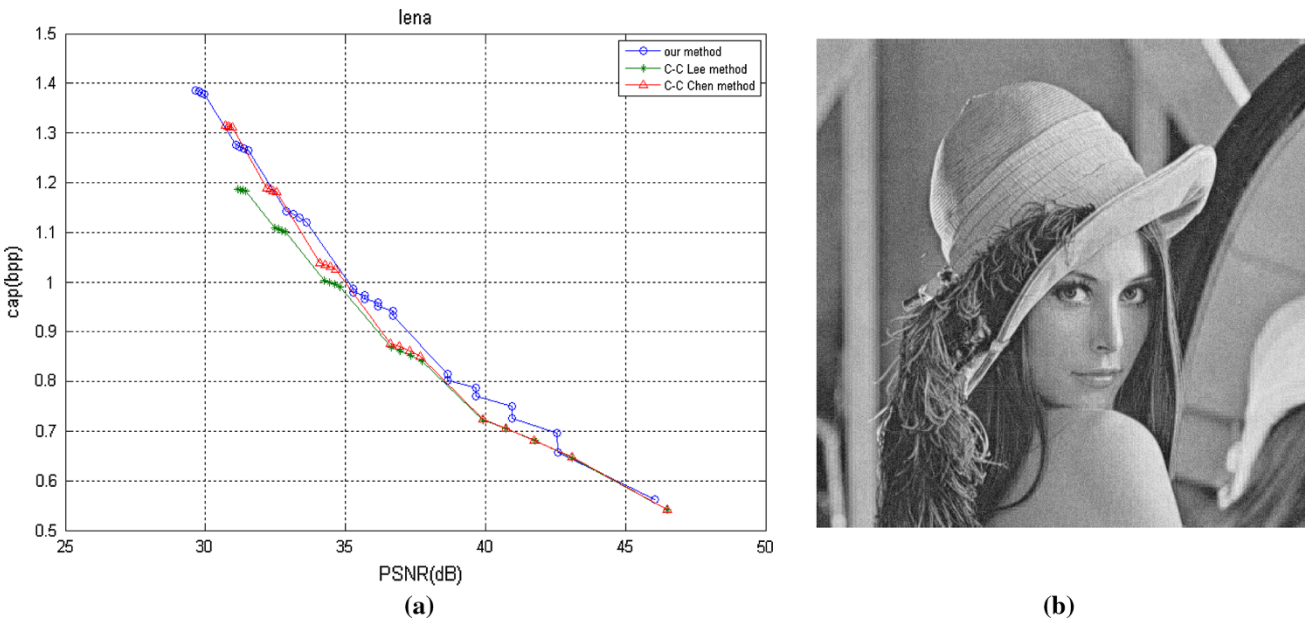


Fig. 10 Results with `Lena`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 383,730 bits (1.4638 bpp). Embedding strength is 48, and PSNR value is 29.6477 dB

pyramidal structure as an example. The image in the lower part of Fig. 4 is the split image in the right part of Fig. 3, which denotes the first layer. By gathering all the reference points together, the second layer can be formed. With the same manner, the points with diagonal lines can form the third layer in the right part of Fig. 4. It implies the pyramidal structure at the right-hand side of Fig. 4. Each layer can be regarded as a new image relating to the original, and data hiding can be performed accordingly with the predetermined embedding strength.

With the arrangements of pyramidal structure of the original image, for the use of the second layer, we may expect the increase of capacity by 25 %. Also, for the third layer, additional increase of $(25\ \%)^2 = 6.25\ \%$ in capacity may also be expected. For the upper layers, they may reside a much fewer capacity with the decreasing rate in a geometric man-

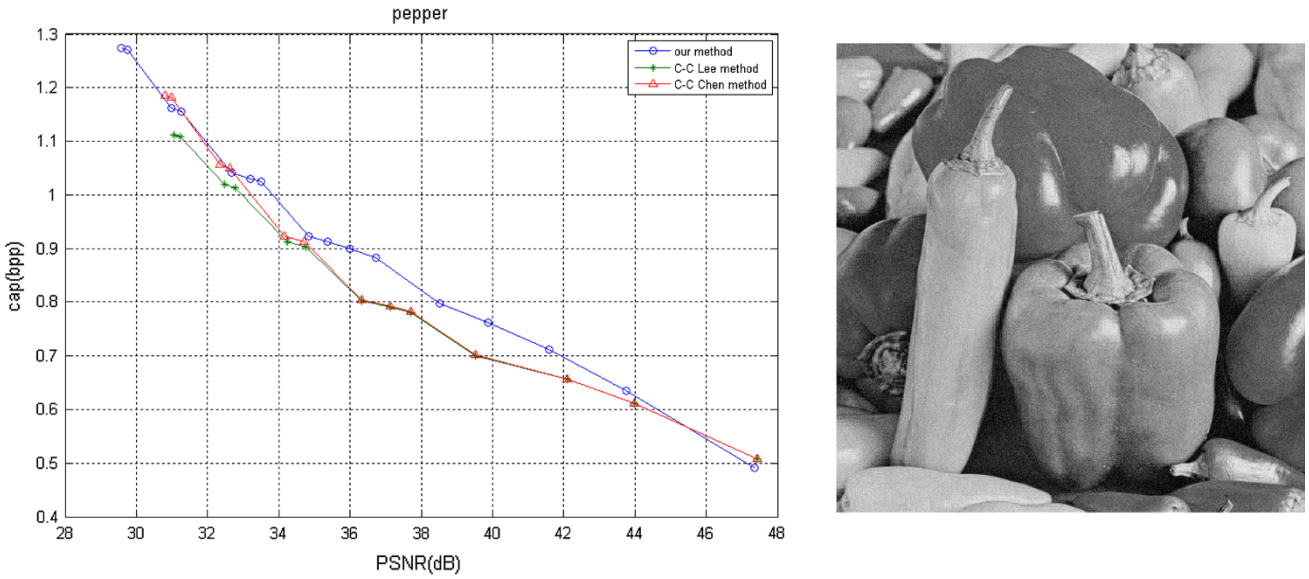


Fig. 11 Results with `pepper`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 354,591 bits (1.3527 bpp). Embedding strength is 48, and PSNR value is 29.5643 dB

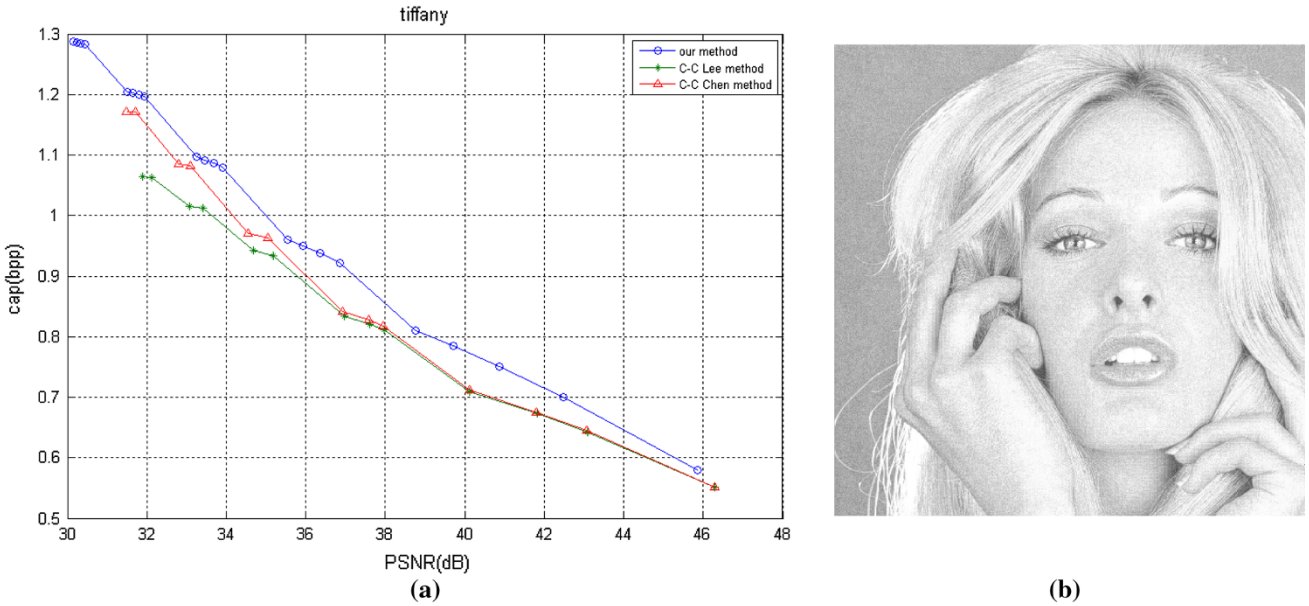


Fig. 12 Results with `Tiffany`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 357,777 bits (1.3648 bpp). Embedding strength is 48, and PSNR value is 30.1508 dB

ner. Considering practical implementation, the use of three layers in pyramidal structure might be a feasible choice for images with sizes of 512×512.

Corresponding to the general framework of reversible data hiding in Fig. 1, we depict the framework of proposed algorithm in Fig. 5 for clarity. On the one hand, in Fig. 5a, at the encoder, pyramidal structure of original image is formed, and reference points are selected. Next, difference values are calculated, and then they are altered for embedding secret information. Finally, both the marked image and side information for decoding are delivered to the decoder. On the other hand, at the decoder in Fig. 5b, procedures are in reverse order to the encoder counterpart. The frameworks in Fig. 5 correspond to the descriptions of proposed algorithm in Sect. 3.

4 Experimental results

In our simulations, we choose the eight test images `baboon, Barbara, boat, F16, Lena, pepper,`

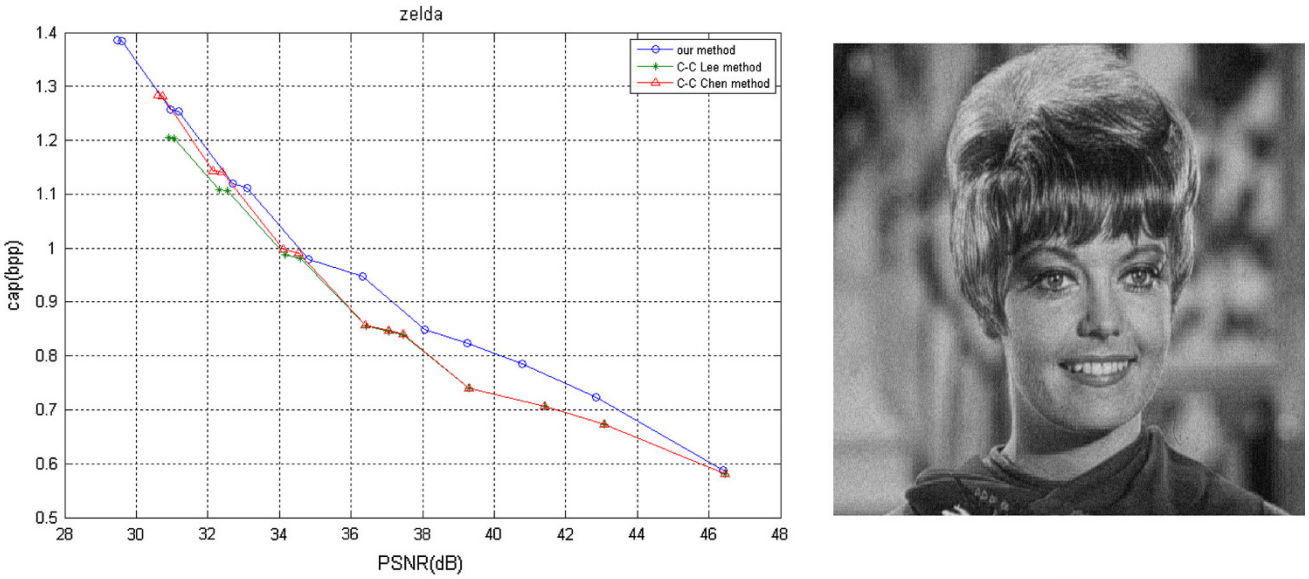


Fig. 13 Results with `Zelda`. **a** Performance evaluation and comparisons with [16] and [17]. **b** Subjective evaluation with the maximally allowable capacity when 383,658 bits (1.4635 bpp). Embedding strength is 48, and PSNR value is 29.4611 dB

`Tiffany`, and `Zelda` with the picture sizes of 512 × 512, for conducting simulations. The secret information to be hidden is the randomly generated bitstreams. Since the proposed method in this paper extends the concepts in [16] and [17], results from the two papers are also compared. Besides, performances with [12] exhibit inferior results than those in [16,17], and results in this paper, thus, we omit to make comparisons with the results in [12].

By properly adjusting the embedding strengths, performances with the eight test images in alphabetical order are depicted in Figs. 6, 7, 8, 9, 10, 11, 12, and 13. In each figure, taking Fig. 6 as an instance, Fig. 6a in the left presents performance comparisons with those in [16] and [17], and Fig. 6b in the right illustrates the subjective image quality for evaluations. For comparing the embedding capacity, we find that with our algorithm, the more amount of secret can be embedded. Among them, the `F16` image can hide at most 408,414 bits (or 1.5580 bpp) in Fig. 9a, and b is depicted for subjective comparisons, with the PSNR of 30.55 dB. We observe that except for the `baboon` image in Fig. 6, our algorithm outperforms that in [16] and [17]. It might be because the `baboon` image displays more active than others, which may lead to the large values in differences. However, for large embedding capacities in `baboon`, we embed more secret with better quality. For the remaining images in Figs. 7, 8, 9, 10, 11, 12 and 13, our algorithm performs better in general. Nevertheless, for low embedding capacities, it performs a bit inferior in `Barbara` in Fig. 7a and `boat` in Fig. 8a. It might be because pyramidal structure brings overhead into data embedding, and it causes degradation to output image

Table 1 Results for `Lena` with $T = 8$ for three-layer adaptive data hiding

Layers	PSNR (dB)	Increase in PSNR (dB)	Capacity (bpp)	Increase in capacity (%)
Layer 1	47.1431	–	0.49457	–
Layers 1 and 2	46.1763	−0.9668	0.61865	25.10
Layers 1, 2 and 3	46.0442	−1.0989	0.64005	29.43

Table 2 Results for `Lena` with $T = 16$ for three-layer adaptive data hiding

Layers	PSNR (dB)	Increase in PSNR (dB)	Capacity (bpp)	Increase in capacity (%)
Layer 1	39.7974	–	0.68904	–
Layers 1 and 2	38.8233	−0.9741	0.86127	25.00
Layers 1, 2 and 3	38.6599	−1.1375	0.89296	29.59

Table 3 Results for Lena with $T = 24$ for three-layer adaptive data hiding

Layers	PSNR (dB)	Increase in PSNR (dB)	Capacity (bpp)	Increase in capacity (%)
Layer 1	36.4280	–	0.81892	–
Layers 1 and 2	35.4465	−0.9815	1.02384	25.02
Layers 1, 2 and 3	35.2756	−1.1524	1.06433	29.97

Table 4 Results for Lena with $T = 32$ for three-layer adaptive data hiding

Layers	PSNR (dB)	Increase in PSNR (dB)	Capacity (bpp)	Increase in capacity (%)
Layer 1	34.0567	–	0.93922	–
Layers 1 and 2	33.0852	−0.9715	1.17380	24.98
Layers 1, 2 and 3	32.9010	−1.1557	1.22111	30.01

Table 5 Results for Lena with $T = 40$ for three-layer adaptive data hiding

Layers	PSNR (dB)	Increase in PSNR (dB)	Capacity (bpp)	Increase in capacity (%)
Layer 1	32.2541	–	1.04117	–
Layers 1 and 2	31.2833	−0.9708	1.30149	25.00
Layers 1, 2 and 3	31.0863	−1.1678	1.35452	30.10

Table 6 Results for Lena with $T = 48$ for three-layer adaptive data hiding

Layers	PSNR (dB)	Increase in PSNR (dB)	Capacity (bpp)	Increase in capacity (%)
Layer 1	30.8220	–	1.12453	–
Layers 1 and 2	29.8602	−0.9618	1.40555	24.99
Layers 1, 2 and 3	29.6477	−1.1743	1.46381	30.17

quality. We are revising our algorithm to conquer the extreme presentation for low capacity.

We also perform the detailed analysis of the results with Lena in Tables 1, 2, 3, 4, 5 and 6. We employ the three-layer pyramid for reversible data hiding. Under a variety of selections of predetermined threshold T, which is an integer with multiples of 8, we observe that if we use more layers for data embedding, then the output image quality gets degraded. We first observe that the increase in capacity is regular; if we use two or three layers for data embedding, the percentage of increase lies around 25 and 31.25 %, respectively. This comes from the observation that the area of the second layer is a quarter of the first later, while the area of the third layer is 1/16 of the first layer. Next, with additional layers for data embedding, the decrease in PSNR values can be expected, from 0.9618 to 0.9815 dB for using two layers, and from 1.0989 to 1.1743 dB for using three layers altogether. The decrease in PSNR comes from the selection of threshold T, and the different characteristics of original images in pyramidal structure. With our method, based on practical requirements, we can predict the necessary capacity with the adaptive embedding with pyramidal structure.

5 Conclusions

In this paper, we presented an adaptive algorithm of reversible data hiding, which employs pyramidal structure of original images for the better capability to hide more secret bits. Reversible data hiding with the alteration of difference values, obtained based on the characteristics of original images, has presented better performances compared to the conventional histogram-based schemes. For reversible data hiding, the reversibility must be retained at the decoder. Then, performances of algorithm, including the output image quality and capacity, can subsequently be examined.

Inspired by scalable coding of multimedia, we can carefully manipulate difference values in the original image between different layers of the pyramidal structure. Adaptive embedding can be applied to one of the four cases with the characteristics of original image. At the encoder, for adaptive embedding with pyramidal structure, performances with our algorithm present better in general for most test images. At the decoder, with the embedding strength, which implies the side information for decoding, the secret information can perfectly be retrieved. In addition, with the aid of location

map, original image can perfectly be recovered. With our algorithm, we can embed more amount of secret with similar output quality. By use of the pyramidal structure, inherent characteristics can be utilized, and better performances can be obtained.

Acknowledgments This work is supported in part by the National Science Council of Taiwan, R.O.C., under Grants NSC102-2220-E-390-002. We would like to thank Mr. S. H. Li for part of the programming practices.

References

1. Petitcolas, F.A.P., Anderson, R.J., Kuhn, M.G.: Information hiding—a survey. Proc. IEEE **87**(7), 1062–1078 (1999)
2. Huang, H.C., Fang, W.C.: Metadata-based image watermarking for copyright protection. Simul. Model. Pract. Theory **18**(4), 436–445 (2010)
3. Ni, Z., Shi, Y.-Q., Ansari, N., Su, W.: Reversible data hiding. IEEE Trans. Circuits Syst. Video Technol. **16**(3), 354–362 (2006)
4. Huang, H.C., Fang, W.C.: Techniques and applications of intelligent multimedia data hiding. Telecommun. Syst. **44**(3–4), 241–251 (2010)
5. Tian, J.: Reversible data embedding using a difference expansion. IEEE Trans. Circuits Syst. Video Technol. **13**(8), 890–896 (2003)
6. Alattar, A.M.: Reversible watermark using the difference expansion of a generalized integer transform. IEEE Trans. Image Process. **13**(8), 1147–1156 (2004)
7. Huang, H.C., Fang, W.C., Lai, W.H.: Secure medical information exchange with reversible data hiding. In: Proceedings of the IEEE International Symposium on Circuits and Systems, pp. 1424–1427 (2012)
8. Fallahpour, M., Megias, D., Ghanbari, M.: Reversible and high-capacity data hiding in medical images. IET Image Process. **5**(2), 190–197 (2011)
9. Zhang, X.: Separable reversible data hiding in encrypted image. IEEE Trans. Inf. Forensics Secur. **7**(2), 826–832 (2012)
10. Feng, G., Fan, L.: Reversible data hiding of high payload using local edge sensing prediction. J. Syst. Softw. **85**(2), 392–399 (2012)
11. Chung, K.L., Huang, Y.H., Yan, W.M., Teng, W.C.: Distortion reduction for histogram modification-based reversible data hiding. Appl. Math. Comput. **218**(9), 5819–5826 (2012)
12. Huang, H.C., Chang, F.C.: Hierarchy-based reversible data hiding. Expert Syst. Appl. **40**(1), 34–43 (2013)
13. Hu, Y., Lee, H.K., Li, J.: DE-based reversible data hiding with improved overflow location map. IEEE Trans. Circuits Syst. Video Technol. **19**(2), 250–260 (2009)
14. Liu, M., Seah, H.S., Zhu, C., Lin, W., Tian, F.: Reducing location map in prediction-based difference expansion for reversible image data embedding. Signal Process. **92**(3), 819–828 (2012)
15. Lee, C.F., Huang, Y.L.: An efficient image interpolation increasing payload in reversible data hiding. Expert Syst. Appl. **39**(8), 6712–6719 (2012)
16. Lee, C.C., Wu, H.C., Tsai, C.S., Chu, Y.P.: Adaptive lossless steganographic scheme with centralized difference expansion. Pattern Recognit. **41**(6), 2097–2106 (2008)
17. Chen, C.C., Tsai, Y.H.: Adaptive reversible image watermarking scheme. J. Syst. Softw. **84**(3), 428–434 (2011)